Please amend the claims as follows:

CLAIMS

1. **(CURRENTLY AMENDED)** A computer implemented method for ~~a primary application to~~ verifying the integrity of a secondary application ~~including~~ comprising the steps of:
   i. obtaining a first reference reduced representation by:
      A. applying a process to obtain a first schema metadata representative of a database structure of a database from the secondary application;
      B. creating ~~a~~ the first reference reduced representation of the first obtained schema metadata using an algorithm; and
      C. storing the first reference reduced representation;
   ii. during execution of ~~the~~ a primary application, applying the process to obtain ~~the~~ second schema metadata representative of the database structure from the secondary application;
   iii. creating a second reduced representation of the second obtained schema metadata using the algorithm;
   iv. comparing the reference reduced representation ~~and~~ with the second reduced representation to provide an indication of the integrity of the second application; and
   v. controlling execution of the primary application dependent on the ~~outcome of the comparison~~ indication.

2. **(CURRENTLY AMENDED)** The method as claimed in claim 1 wherein the secondary application is ~~a~~ the database.

3. **(CURRENTLY AMENDED)** The method as claimed in claim 2 wherein the first or second schema metadata is selected from the set of tables, columns in tables, data types of columns, lengths of columns, custom database data types, foreign keys, constraints, stored procedures, views, triggers, indices, and scheduled jobs.

4. **(PREVIOUSLY PRESENTED)** The method as claimed in claim 2 wherein the algorithm is a hash function.

5. **(PREVIOUSLY PRESENTED)** The method as claimed in claim 3 wherein the hash function is one selected from the set of MD5 and CRC32.

6. **(PREVIOUSLY PRESENTED)** The method as claimed in claim 2 wherein the algorithm is a lossless compression algorithm.

7. **(PREVIOUSLY PRESENTED)** The method as claimed in claim 6 wherein the lossless compression algorithm is one selected from the set of zip, gzip, and bzip2.

8. **(CURRENTLY AMENDED)** The method as claimed in claim 2 wherein the first reference reduced representation is stored by embedding the representation within the primary application.

9. **(CURRENTLY AMENDED)** The method as claimed in claim 2 wherein the first reference reduced representation is stored by embedding the representation within configuration files for the primary application.

10. **(PREVIOUSLY PRESENTED)** The method as claimed in claim 2 wherein step (i) is repeated before steps (ii) to (v) at least one time when an expected change occurs to the schema metadata in the database.

11. **(PREVIOUSLY PRESENTED)** The method as claimed in claim 2 wherein the process includes organizing the extracted schema metadata using a nested and determinable method.

12. **(PREVIOUSLY PRESENTED)** The method as claimed in claim 11 wherein the nested and determinable method is by alphabetical listing of the schema metadata elements.

13. **(PREVIOUSLY PRESENTED)** The method as claimed in claim 11 wherein the nested and determinable method is by default database order of the schema metadata elements.

14. **(PREVIOUSLY PRESENTED)** The method as claimed in claim 11 wherein the nested and determinable method is by creation date order of the schema metadata elements.

15. **(PREVIOUSLY PRESENTED)** The method as claimed in claim 11 wherein the nested and determinable method is by table owner of the schema metadata elements.

16. **(PREVIOUSLY PRESENTED)** The method as claimed in claim 2 wherein the execution of the primary application is controlled by halting execution of the primary application.

17. **(PREVIOUSLY PRESENTED)** The method as claimed in claim 2 wherein the execution of the primary application is controlled by the primary application sending an error message to one selected from the set of a user of the primary application, a manager of the primary application, a manager of the database, and the database.

18. **(CURRENTLY AMENDED)** The method as claimed in claim 2 ~~including~~ further comprising the step of:

    i. requesting a schema stability lock of the database.

19. **(PREVIOUSLY PRESENTED)** The method as claimed in claim 2 wherein the process obtains all available schema metadata.

20. **(PREVIOUSLY PRESENTED)** The method as claimed in claim 2 wherein the process only obtains the schema metadata which would affect the primary application if that schema metadata were to change.

21. **(PREVIOUSLY PRESENTED)** The method as claimed in claim 2 wherein the process utilizes SQL92 standard to obtain the schema metadata from the database.

22. **(PREVIOUSLY PRESENTED)** The method as claimed in claim 2 wherein the process utilizes the database's API to obtain the schema metadata from the database.

23. **(PREVIOUSLY PRESENTED)** The method as claimed in claim 22 wherein the database's API is a Java database API.

24. **(CURRENTLY AMENDED)** A system for verifying ~~for a plurality of applications~~ the integrity of one or more databases ~~including~~ comprising:
    i. a plurality of applications adapted to store a plurality of previously calculated reduced representations of schema metadata representative of the structure of ~~for~~ one or more databases, to extract a plurality of schema metadata representative of database structure from one or more databases, to newly calculate a plurality of reduced representations from the plurality of extracted schema metadata, and to compare each of plurality of previously calculated reduced representations with its corresponding newly calculated reduced representation to provide an indication of the integrity of one or more databases; and
    ii. one or more databases adapted to receive requests for schema metadata from the plurality of applications and to transmit schema metadata to the plurality of applications dependent of the indication.

25. **(PREVIOUSLY PRESENTED)** The system as claimed in claim 24 wherein the schema metadata is selected from the set of tables, columns in tables, data types of columns, lengths of columns, custom database data types, foreign keys, constraints, stored procedures, views, triggers, indices, and scheduled jobs.

26. **(PREVIOUSLY PRESENTED)** The system as claimed in claim 24 wherein the reduced representations are calculated using a hash function.

27. **(PREVIOUSLY PRESENTED)** The system as claimed in claim 26 wherein the hash function is one selected from the set of MD5 and CRC32.

28. **(PREVIOUSLY PRESENTED)** The system as claimed in claim 24 wherein reduced representations are calculated using a lossless compression algorithm.

29. **(PREVIOUSLY PRESENTED)** The system as claimed in claim 28 wherein the lossless compression algorithm is one selected from the set of zip, gzip, and bzip2.

30. **(PREVIOUSLY PRESENTED)** The system as claimed in claim 24 wherein each previously calculated reduced representation is stored by embedding the representation within its associated application.

31. **(PREVIOUSLY PRESENTED)** The system as claimed in claim 24 wherein each previously calculated reduced representation is stored by embedding the representation within configuration files for its associated application.

32. **(PREVIOUSLY PRESENTED)** The system as claimed in claim 24 wherein each schema metadata is organized using a nested and determinable method before its reduced representation is calculated.

33. **(PREVIOUSLY PRESENTED)** The system as claimed in claim 32 wherein the nested and determinable method is by alphabetical listing of the schema metadata elements.

34. **(PREVIOUSLY PRESENTED)** The system as claimed in claim 32 wherein the nested and determinable method is by default database order of the schema metadata elements.

35. **(PREVIOUSLY PRESENTED)** The system as claimed in claim 32 wherein the nested and determinable method is by creation date order of the schema metadata elements.

36. **(PREVIOUSLY PRESENTED)** The system as claimed in claim 32 wherein the nested and determinable method is by table owner of the schema metadata elements.

37. **(PREVIOUSLY PRESENTED)** The system as claimed in claim 24 wherein the result of each comparison controls execution of its associated application

38. **(PREVIOUSLY PRESENTED)** The system as claimed in claim 37 wherein the execution of the application is controlled by halting execution of the application.

39. **(PREVIOUSLY PRESENTED)** The system as claimed in claim 37 wherein the execution of the application is controlled by the application sending an error message to one selected from the set of a user of the application, a manager of the application, a manager of the associated database, and the associated database.

40. **(PREVIOUSLY PRESENTED)** The system as claimed in claim 24 wherein the plurality of applications are further adapted to request a schema stability lock of the one or more databases.

41. **(PREVIOUSLY PRESENTED)** The system as claimed in claim 24 wherein each application is adapted to extract all available schema metadata from each database.

42. **(PREVIOUSLY PRESENTED)** The system as claimed in claim 24 wherein each application is adapted to extract the schema metadata which would affect the application if that schema metadata were to change.

43. **(PREVIOUSLY PRESENTED)** The system as claimed in claim 24 wherein each application is adapted to utilize SQL92 standard to extract the schema metadata from each database.

44. **(PREVIOUSLY PRESENTED)** The system as claimed in claim 24 wherein each application is adapted to utilize the database's API to extract the schema metadata from each database.

45. **(PREVIOUSLY PRESENTED)** The system as claimed in claim 44 wherein the database's API is a Java database API.

46. **(CURRENTLY AMENDED)** A system for verifying ~~for an application~~ the integrity of a database ~~including~~ comprising:
    i. an application;
    ii. a stored reduced representation of schema metadata representative of the structure of ~~a~~ the database; and
    iii. a verification engine which upon connection to ~~a~~ the database obtains a reduced representation of schema metadata representative of the structure of the database from the database and compares it with the stored reduced representation in order to provide an indication of the integrity of the database to control the application based upon the indication.

47. **(PREVIOUSLY PRESENTED)** The system as claimed in claim 46 wherein the schema metadata is selected from the set of tables, columns in tables, data types of columns, lengths of columns, custom database data types, foreign keys, constraints, stored procedures, views, triggers, indices, and scheduled jobs.

48. **(PREVIOUSLY PRESENTED)** The system as claimed in claim 46 wherein the reduced representations are calculated using a hash function.

49. **(PREVIOUSLY PRESENTED)** The system as claimed in claim 46 wherein the stored reduced representation is stored by embedding the representation within the application.

50. **(PREVIOUSLY PRESENTED)** The system as claimed in claim 48 wherein each schema metadata is organized using a nested and determinable method before its reduced representation is calculated.

51. **(PREVIOUSLY PRESENTED)** The system as claimed in claim 46 wherein the application is controlled by halting execution of the application.

52. **(PREVIOUSLY PRESENTED)** The system as claimed in claim 46 wherein the application is controlled by the application sending an error message to one selected from the set of a user of the application, a manager of the application, a manager of the associated database, and the associated database.

53. **(CANCELLED)**

54. **(CANCELLED)**

55. **(ORIGINAL)** Storage media containing software as claimed in claim 54.